REMARKS

The present application was filed on June 20, 2003 with claims 1 through 21. Claims 1 through 21 are presently pending in the above-identified patent application.

In the Office Action, the Examiner rejected claims 1-9, 14-16, 20, and 21 under 35 U.S.C. §102(e) as being anticipated over Moir (United States Patent Publication Number 2002/120720), and rejected claims 10-14 under 35 U.S.C. §103(a) as being unpatentable over Moir in view of Tuatini (United States Patent Publication Number 2001/0047385). The Examiner also rejected claims 17-19 under 35 U.S.C. 103(a) as being unpatentable over Moir and in view of Presley (United States Patent Number 2003/0105838).

Independent Claims 1, 20 and 21

Independent claims 1, 20, and 21 were rejected under 35 U.S.C. §102(e) as being anticipated by Moir. Regarding claim 1, the Examiner asserts that Moir discloses generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements (page 5, paragraph 58)); and generating at least one executable module adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element (page 5, paragraph 60).

Applicants note that the present specification teaches that

> *input rules* are also part of specifications for a device and comprise, for example, *a set of checks or constraints or both* that should be performed before or after a configuration element is accessed. The input rules are generally derived from 'domain experts' (typically network specialists). An input rule is *usually represented as a set of executable statements*. (Page 2, lines 24-28.)

The present specification also teaches that

> *output rules* are *determined by using the accessed configuration elements, the input rules, and the way the input rule manipulates its accessed configuration elements*. Regarding the latter, output rules may be determined to deal with modifications to configuration elements, as explained in more detail below. In an illustrative embodiment, *each output rule is generally derived from exactly one input rule and corresponds to the same input configuration element associated with that input rule*. Output rules may be derived from multiple input rules, if desired. (Page 3, lines 7-14.)

Finally, the present disclosure teaches that

> an *executable module* is generated that is *adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element*. (Page 3, lines 15-17.)

In the text cited by the Examiner, however, Moir teaches:

> [0058] *The virtual machine compiler 60 utilizes the operations file 62 and the rule file 64 to compile a rule program 66*, which in one embodiment comprises a binary object including a sequence of instructions suitable for the virtual machine 10, discussed above. The rule program 66 comprises a set of operations, selected from operations supported by components of the network connection device 12, for performance by the respective components of the network connection device in accordance with the behavioral requirements defined by the rule file 64. In one embodiment, the rule program 66 may embody a number of sequences, these sequences constituting the classification rules 18, the event management rules 17 and the label management rules 19 discussed above with reference to FIG. 3.
> [0059] The virtual machine compiler 60 is accordingly used to define the behavior of a virtual machine 10 in a secure and performance-oriented manner by loading the rule program 66 into the key locations of the virtual machine 10.
> [0060] The virtual machine compiler 60, in one embodiment, presents a model to a rule designer that consists of a number of abstract data processes and contexts, as illustrated in FIG. 10. Specifically, FIG. 10 illustrates the rule program 66 as conceptually comprising a number of rules 68 (i.e., instruction sequences) that are utilized to bind process behavior definitions, conveniently labeled operations 70, to contextual zed sets of data, conveniently labeled registers 72. It will be appreciated that because a specific network connection device 12 may be constituted by a number of smaller components, the overall process and context for a network connection device 12 may similarly viewed as constituting a number of corresponding components. As illustrated in FIG. 10, each component (e.g., the TCP protocol or an ATM device driver) that wishes to contribute to a process (e.g., an abstract entity such as a data plane or the management plane) can operate, via a rule 68 class on a new or existing register 72.
> (Emphasis added.)

In the present Office Action, the Examiner asserts that paragraph 58 teaches that "the rule program is derived by compiling the rule file and operations file" (Office Action: page 3) and asserts that "the compiler binds processes behavior definitions and operations to data through compiling the operations file and rule files" (Office Action: page 10). Moir teaches, however, that the "*virtual machine compiler 60 utilizes the operations file 62 and the rule file 64 to compile a rule program 66*." (Emphasis added.) The Examiner has changed this statement to assert that Moir teaches "*compiling the operations file and rule files*." The statement "compiling

a rule file" has a different meaning than "compiling a rule program." The Examiner's statement implies that the rule file contains executable statements; the teaching in Moir does *not* imply that the rule file contains executable statements. Moir does *not* teach that the rule file is compiled and does *not* disclose or suggest that *one or more input rules comprise one or more* ***executable statements***. Independent claims 1, 20, and 21 require *wherein said one or more input rules comprise one or more* ***executable statements***.

Thus, Moir does not disclose or suggest generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements; and generating at least one executable module adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element, as required by independent claims 1, 20, and 21.

Additional Cited References

Tuatini was also cited by the Examiner for its disclosure of generating at least one class for a given one of the one or more output rules. Tuatini does *not*, however, address the issue of generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements.

Thus, Tuatini does not disclose or suggest generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements; and generating at least one executable module adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element, as required by independent claims 1, 20, and 21.

Presley was also cited by the Examiner for its disclosure of performing a circularity check by determining dependency relationships between the two or more output rules and by determining whether a given one of the two or more output rules depends upon itself.

Presley does *not*, however, address the issue of generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements.

Thus, Presley does not disclose or suggest generating one or more output rules using at least the accessed information, the accessed configuration elements, and the input rules, wherein an output rule corresponds to one or more input configuration elements and wherein said one or more input rules comprise one or more executable statements; and generating at least one executable module adapted to access at least a given one of the input configuration elements and to trigger one or more of the output rules corresponding to the given input configuration element, as required by independent claims 1, 20, and 21.

Dependent Claims 2-19

Claims 2-19 are dependent on claim 1 and are therefore patentably distinguished over Moir, Tuatini, and Presley, alone or in combination, because of their dependency from independent claim 1 for the reasons set forth above, as well as other elements these claims add in combination to their base claim.

If any outstanding issues remain, or if the Examiner has any further suggestions for expediting allowance of this application, the Examiner is invited to contact the undersigned at the telephone number indicated below.

The Examiner's attention to this matter is appreciated.

Respectfully submitted,

/Kevin M. Mason/

Date: December 29, 2008

Kevin M. Mason
Attorney for Applicant(s)
Reg. No. 36,597
Ryan, Mason & Lewis, LLP
1300 Post Road, Suite 205
Fairfield, CT 06824
(203) 255-6560